CIV 2552 – Mét. Num. Prob. de Fluxo e Transporte em Meios Porosos
1º Semestre – 2010

Trab3: Método dos Elementos Finitos
Fluxo hidráulico em regime permanente 1D

Resolva a 1ª Questão do segundo trabalho para regime permanente pelo Método dos Elementos Finitos utilizando elementos lineares. Complete o programa em MATLAB onde indicado.

trab3.m

```
% Include global variables
include_gblrefs;

% Preprocessing
preprocessor;

% Assembly global permeability matrix Kg
% COMPLETE AQUI

% Assembly global forcing vector F
% COMPLETE AQUI

% Solve for u
% u = Kg \ F;

% Plot steady-state solution
postprocessor;
```

include_gblrefs.m

```
% File to include global variables

% Global size parameters
global nnp                        % number of nodal points
global nel                        % number of elements

% Nodal coordinates
global x                          % array of nodal x coordinates

% Element properties arrays
global CArea                      % array of element cross-sectional area values
global Permeability               % array of element permeability values

% Element nodal connectivity array, location matrix and gather matrix
global LM                         % gather matrix: LM(nen,nel)
                                  %  stores global nodal number for
                                  %  each local node of each element

% Global equation matrix, forcing vector, and solution vector
global Kg                         % global permeability matrix
global F                          % global system forcing vector
global u                          % global system solution vector
```

```
% Boundary Conditions (B.C.) information
%  flag = 0 --> natural B.C.
%  flag = 1 --> essencial B.C.
global bc_init_flag            % initial node B.C. flag
global bc_end_flag             % end node B.C. flag
global bc_init_val             % initial node B.C. value
global bc_end_val              % end node B.C. value

% Distributed and point source loads
global dist_load               % array of element distributed source loads

% Analytical soluction data
global nasp                    % number of analytical solution points
global x_as                    % x coordinates of analytical solution points
global u_as                    % analytical field solution values
global q_as                    % analytical flux solution values
```

---

preprocessor.m

```
% Preprocessor:
% input data for 1D example of Trab2 with linear elements

function preprocessor

include_gblrefs;

% 1D domain parameters
L     = 80.0;            % length [m]
A     = 5.0;             % cross-section area [m2]
K     = 8.0e-6;          % permeability coefficient [m/s]
q     = 1.0e-6;          % distributed external source load [m/s]
H1    = 40.0;            % essencial B.C. at beginning (x = 0) [m]
H2    = 5.0;             % essencial B.C. at end (x = L) [m]

% Discretization parameters
nnp  =  6;               % number of nodal points
nel  =  5;               % number of elements

% Global equations coefficient matrices, RHS vector, and solution vector
Kg  = zeros(nnp,nnp);    % initialize global permeability matrix
F   = zeros(nnp,1);      % initialize global system forcing vector
u   = zeros(nnp,1);      % initialize global system solution vector

% Element properties vectors
CArea          = A*ones(nel,1);  % cross-section area
Permeability   = K*ones(nel,1);  % permeability coefficient

% B.C.'s
bc_init_flag = 1;
bc_end_flag  = 1;
bc_init_val  = H1;
bc_end_val   = H2;

% Element distributed source load vector
dist_load = ones(nel,1)*q;
```

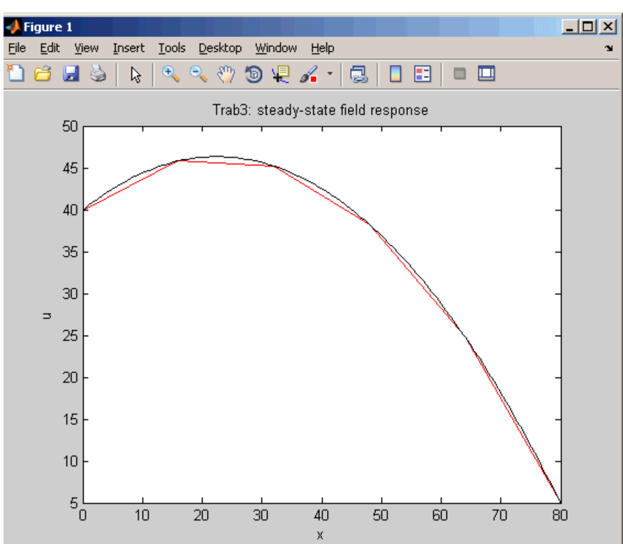
```
% x coordinates array
x = zeros(nnp,1);
x = linspace(0.0,L,nnp);

% gather matrix = connectivity array
LM = zeros(2,nel);
LM(1,:) = (1:1:nnp-1);
LM(2,:) = (2:1:nnp);

% Steady-state analytical solution
nasp = 101;
x_as = zeros(nasp,1);
x_as = linspace(0.0,L,nasp);
for i=1:nasp
 u_as(i) = -0.0125*x_as(i)^2 + 0.5625*x_as(i) + 40.0;
 q_as(i) = -K * (-0.025*x_as(i) + 0.5625);
end
```

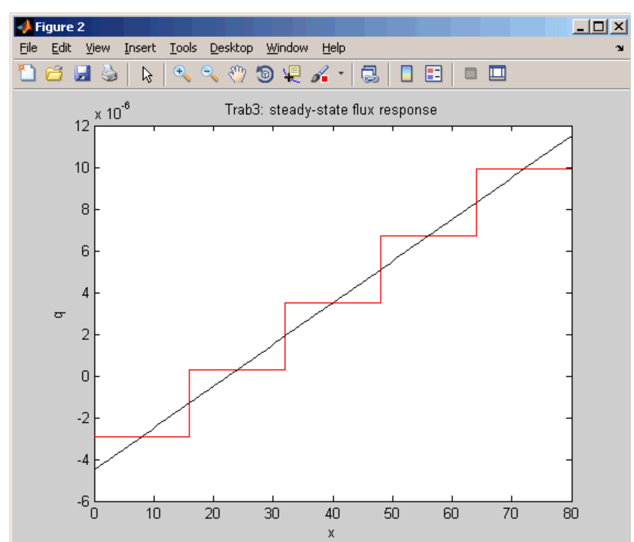
postprocessor.m

```
% Postprocessing steady state plots for given solution vector
% and computed flux

function postprocessor

include_gblrefs;

% Create figure for main field plots and get handle to it
fig_field = figure;

% Locate main field figure at left side of screen
screen_sizes = get(0,'ScreenSize');
fig_field_pos = get( fig_field, 'Position' );
fig_field_pos(1) = 0;
set( fig_field, 'Position', fig_field_pos );

% Plot main field response
plot(x,u,'Color','r');

% Setup labels
xlabel('x');
ylabel('u');
title('Trab3: steady-state field response');
hold on

% Dimension flux response arrays
x_flux = zeros(nel*2,1);
v_flux = zeros(nel*2,1);

% Compute flux response from given solution vector and plot it
for e=1:nel
 ni  = LM(1,e);                 % initial node of element
 nj  = LM(2,e);                 % final node of element
 le  = x(nj)-x(ni);             % compute element length
 ui  = u(ni);                   % initial element node field value
 uj  = u(nj);                   % final element node field value
 K   = Permeability(e);         % get element permeability coefficient
```

```
 x_flux(2*e-1) = x(ni);         % first flux point in element is located
                                %  at first element node
 x_flux(2*e)   = x(nj);         % second flux point in element is located
                                %  at last element node
 v_flux(2*e-1) = -K * (uj-ui)/le;
 v_flux(2*e)   = -K * (uj-ui)/le;
end

% Create figure for flux response plots and get handle to it
fig_flux = figure;

% Locate flux results figure at right side of screen
screen_sizes = get(0,'ScreenSize');
fig_flux_pos = get( fig_flux, 'Position' );
fig_flux_pos(1) = screen_sizes(3) - fig_flux_pos(3);
set( fig_flux, 'Position', fig_flux_pos );

% Plot given solution vector
plot(x_flux,v_flux,'Color','r');

% Setup labels
xlabel('x');
ylabel('q');
title('Trab3: steady-state flux response');
hold on

% Plot analytical solutions (if available)
if( nasp )
 figure( fig_field );
 plot(x_as,u_as,'Color','k');
 figure( fig_flux );
 plot(x_as,q_as,'Color','k');
end
```

Imagens dos resultados (regime permanente) obtidos para o exemplo da 1ª Questão do segundo trabalho utilizando 5 elementos finitos lineares:





Consideração das condições de contorno essenciais (de Dirichlet)

Uma maneira conveniente para considerar as condições de contorno essenciais do problema proposto ($h_1 = H_1$ e $h_n = H_n$) é modificar a matriz global de permeabilidade $[Kg]$ e o vetor das cargas equivalentes nodais $\{F\}$ depois de eles terem sido criados sem considerar nenhuma condição de contorno. Na formulação do problema 1D com o elemento finito linear com dois nós, a matriz $[Kg]$ tem um formato tridiagonal:

$$\begin{bmatrix} Kg_{1,1} & Kg_{1,2} & 0 & 0 & ... & 0 & 0 \\ Kg_{2,1} & Kg_{2,2} & Kg_{2,3} & 0 & ... & 0 & 0 \\ & & ... & & & & \\ & & & ... & & & \\ 0 & 0 & ... & 0 & Kg_{n-1,n-2} & Kg_{n-1,n-1} & Kg_{n-1,n} \\ 0 & 0 & ... & 0 & 0 & Kg_{n,n-1} & Kg_{n,n} \end{bmatrix} \cdot \begin{Bmatrix} H_1 \\ h_2 \\ ... \\ ... \\ h_{n-1} \\ H_n \end{Bmatrix} = \begin{Bmatrix} F_1 \\ F_2 \\ ... \\ ... \\ F_{n-1} \\ F_n \end{Bmatrix}$$

Apenas as duas primeiras linhas e as duas últimas linhas de $[Kg]$ e de $\{F\}$ precisam são modificadas, tal como indicado abaixo:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & ... & 0 & 0 \\ 0 & Kg_{2,2} & Kg_{2,3} & 0 & ... & 0 & 0 \\ & & ... & & & & \\ & & & ... & & & \\ 0 & 0 & ... & 0 & Kg_{n-1,n-2} & Kg_{n-1,n-1} & 0 \\ 0 & 0 & ... & 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{Bmatrix} h_1 \\ h_2 \\ ... \\ ... \\ h_{n-1} \\ h_n \end{Bmatrix} = \begin{Bmatrix} H_1 \\ F_2 - Kg_{2,1} \cdot H_1 \\ ... \\ ... \\ F_{n-1} - Kg_{n-1,n} \cdot H_n \\ H_n \end{Bmatrix}$$

A primeira e a última linha da matriz ficam com um “1” na diagonal principal e “0” nos outros termos. Os termos de carga no vetor $\{F\}$ na primeira e na última linha têm o valor das condições de contorno essenciais H_1 e H_n, respectivamente. Para manter a simetria da matriz global, o primeiro termo da segunda linha e o último termo da penúltima linha da matriz são anulados, sendo que os termos de carga correspondentes são alterados tal como indicado, levando-se em conta que os termos anulados da matriz são os que multiplicam os valores conhecidos das condições de contorno essenciais. Dessa forma, o número de equações do sistema não se altera em relação ao número total de nós, h_1 e h_n continuam sendo incógnitas, e a solução da primeira e última linhas do sistema resulta nas condições de contorno essenciais.